Coalition formation for egoistic agents

Guillaume Vauvert – Amal El Fallah Seghrouchni
Laboratoire d'Informatique de Paris Nord
Université Paris XIII
Avenue Jean-Baptiste Clément
93430 Villetaneuse
{guillaume.vauvert,amal.elfallah}@lipn.univ-paris13.fr

Abstract

Coalitions are organizations which provide dynamics and flexibility to grasp situations of cooperation and competitiveness in open environments. In an economic distributed context, agents are necessarily heterogeneous, egoistic and free in behavior and in reasoning what brings us to propose a protocol which privileges freedom of agents. Applied to task repartition between agents, this protocol is open, distributed and egalitarian. It is based on exchange of preferences computed using qualitative and quantitative criteria and allows different forms of rationality and strategies. We are going to show that it ends and it guarantees to reach a consensus.

1 Introduction

Main interest of multi-agents systems (M.A.S.) results from collaboration of agents. Several approaches has been studied to support this collaboration : distributed resolution of problems, contractual networks, organization based approaches, protocols of negotiation, coalition formation, *etc.*
Coalitions allow agents to satisfy needs requiring synergy for competences of several agents as, for example, within the framework of the resolution of complex tasks for which agents acting alone would be unable or much less effective.
A coalition can be defined like a short-term organization based on specific and contextual engagements thus making possible agents to benefit from their respective competences (in economic context for example, several companies gather "virtually" to answer to bids requiring various competences). Coalition formation offers several advantages : 1) the concept of punctual engagement allows agents to react in an opportunist way and to dynamically revise their interests and consequently their objectives; 2) the coalitions formation and dissolution are context-dependent, thus they allow agents to dynamically adapt their dealings; 3) contrary to a static organization (*i.e.* preset), coalitions formation allows to apprehend in a more flexible way open and dynamic environments.
From the point of view of M.A.S., researches in coalitions formation try to propose protocols of automation which are more "realistic". Indeed, the search of optimal solutions (generally measured in term of maximum total profit) require NP-complete algorithms and because of this complexity, several simplifying assumptions are introduced in order to generate solutions computable in moderate time. One of the strongest assumptions consists in supposing that agents are co-operative even altruistic. Within the framework of this paper, contrary to existing approaches (cf section 2) and accordingly with a more general sense, we make no assumption on predispositions of agents to collaborate. We are satisfied to motivate them to do it (*e.g.* agents will be brought to cooperate if they find a certain interest there). Thus, each agent is free to behave and is likely to have egoistic objectives (*e.g.* within the framework of electronic commerce). However a set of egoistic agents generally leads to blocking, which finally satisfies nobody. To avoid this type of situations, we propose a protocol for the coalitions formation which respects the freedom of agents while guaranteeing to reach a consensus.
This article is organized as follows: section 2 presents a short state of the art on coalition formation and introduces our approach, in particular, compared to those which deal with egoistic agents. Section 3 justifies our step which privileges the freedom of agents and defines its outlines. Section 4 formalizes concepts of resolution to the problem of consensus within the framework of free agents, while section 5 presents procedures, introducing concepts necessary to the definition of the protocol

of consensus. Section 6 gives the algorithm. Section 7 discusses algorithmic aspects of our protocols like termination, complexity and some aspects of implementation. Finally, section 8 concludes this article and presents prospects.

2 Positioning

Several work in M.A.S. were interested in dynamic organizations and in particular in coalitions formation. The majority of them suppose *a priori* that all agents have the same rationality, which makes it possible to reduce the space of search considerably. The strongest assumption consists in supposing that agents are altruistic, which leads, as in [4], to calculate in a distributed way the solution which maximizes the common utility.

Works with which we compare this paper relates to only egoistic agents, because considering altruistic agents radically changes the problem and thus solutions. In [5], the problem is simplified to calculate in a distributed way the best solution for the system as a whole. It's a difficult problem, because it is not obvious to have a total sight of the system for one of its components. For us, it is a question of managing to reconcile the inevitably conflict individual interests (if not it would not have there problem) to arrive, despite everything, with a solution which is accepted by all, but within the meaning of the legitimacy of its construction.

The problem of coalition formation consists in seeking a solution which is most satisfactory for the set of the agents. It is typically the case when one seeks to calculate the Shapley value (which corresponds roughly to the expectancy of the utility) : one seeks to find the configuration who satisfies overall more the agents. Admittedly, the total interest of M.A.S. is obvious but the imposed method keep the personal freedom of the agents in check. Indeed, either the configuration is calculated in an external way, or it is calculated by the agents themselves, but in giving them a definite role which they must follow. It is the case for example in [6] which proposes a distributed computation process of the Shapley value. The authors impose on the agents that the best solution is the one which maximizes the Shapley value. Admittedly, to guarantee a relative satisfaction of the agent, they take care not to force them to form a coalition which does not bring to him more than what it would have had before. But this deterministic process prevent the agents from developing their own strategies : if an agent agrees to take the risk to create a coalition with poor yield in the hope to form a very productive coalition later, that can be very beneficial for him.

In [2], Steven Ketchpel gave a little more freedom to the agents while allowing them to make different estimations of the income of a coalition. The main problem is the distribution of the benefits which are not known with precision at the instant of the formation. The proposed algorithm consists of a succession of aggregations of agents and coalitions. What is interesting here is that the agents can have different hopes from the incomes and manage an agreement all the same. But, like much of others, this algorithm supposes that all is convertible into currency, that all can be considered by a real value, which has consequences on the algorithm itself, thus returning it very dependent on this assumption. This it the reason to choice the preferences exchanges which are independent of the criteria which produced them, allowing the agents to work out their own strategies in full freedom .

A similar problem has been studied by Kenneth Arrow in [1] : the combination of individual preferences in a collective one. He shows the impossibility to find a vote procedure which respects his five intuitive axioms and which is not dictatorial. The algorithm we present here doesn't respect these axioms, but the procedure try to be the least dictatorial.

The second criticism relates to the dependences between the coalitions such as defined in [2] : in an iterative formation, an agent cannot take into account the possible future participation of other agents ; indeed, the interest for a coalition depends on the set of its members and not only on the increase in its value at the stage in progress. In our approach, all solutions being estimated overall and in parallel, the agents can fully estimate their interest for each solution.

Agents are free, which enables us to take into account naturally what often seems constraints : dependence of tasks and then of coalitions. The protocol is based only on preferences, it does not take account of the reasoning which generated them and consequently, each agent is free to have its own rationality (cf section 3.1). Indeed, a good rationality from a global point of view (of coalition, altruism) is not necessarily good individually and can discourage agents (all or some) to apply it. Thus, it is no more possible to use these presuppositions to calculate the best solution in a context of egoistic agents. In the same way, an individual strategy cannot be always optimal because it depends on the others ones.

3 Our step : to privilege the agents's freedom

3.1 Freedom and rationality

The difficulty of coalitions management increases considerably when one considers agents with freedom to reason and to act. The freedom of reasoning prevents us from making assumptions on knowledge, motivations, mechanisms of inference, or decision of agents. Freedom to act implies that there is not action prohibited *a priori* .
However, freedom is legitimated even essential for a broad class of applications. For example, one cannot suppose that all agents that take part in a trade talk will follow the imposed protocol, nor that they would be ready to reveal their own strategies.
Such a type of agent has obviously many consequences. For example, one cannot make sure any more that a consensus will be reached, because if all agents are against, no protocol will make it possible to reach a state of balance. In the same way, the possibility of cheating, *i.e.* not to follow the protocol, must also be taken into account.
The protocol we present is however more efficient if agents have a particular rationality, but it works even if they have different one. This economic rationality leads agents to make concessions, because if they don't, they'll win less.

3.2 Effects on the protocols and the organizations

The freedom of agents has direct consequences on the design of protocols and organizations of agents in coalitions. To obtain coherent collective behaviors, it is necessary to lay down rules, but to respect personal freedoms, these rules should constraint only *perceptible* datas, *i.e.* behaviors directed towards the others. In the event of fraud, agents can always be sanctioned ; sanctions incite them to follow the given rules.
In such a context, a protocol must be : 1) universal, in particular it assumes nothing on the individual choices, nor on individual rationalities, thus respecting the agents's freedom ; 2) egalitarian, it doesn't support particular classes of agents ; 3) distributed, because centralization brings the usual problems (overload of the network and the central agent, weakness with the breakdowns...) ; in addition, centralization increases the power of the central agent enabling him to cheat ; although one can make the assumption of an impartial agent, the integrity of an agent could always be blamed (external influences, corruption), which would harm the legitimacy of the solution.
The freedom of agents modifies also the perception which one can have of the organizations, which are indeed sometimes regarded as entities by themselves, depending certainly on their members, but having nevertheless own attributes like a certain amount of autonomy. In the context of this article, an organization is a set of dependent agents (by contract, punctually), but which always have all their freedom. Each agent plays a certain role there, but for the same reasons, role and behavior aren't identical. A role is a set of rights and duties, codified in a rule, in the hope to obtain a coherent overall behavior. The rights are represented by a set of possible actions at each stage and a duty by a set of actions awaited by the other agents. But any agent keeps its freedom and can thus respect or not the rule, and it is thus necessary to take into account this possibility.

3.3 Objective : resolution of consensus problem for tasks allocation

We chose the framework of tasks resolution by agents : a system receives a set of tasks divided in sub-tasks and proposes these sub-tasks to registered agents. They try to distribute the sub-tasks among themselves, each one being able to carry out only one part of it. A task is carried out when all its sub-tasks are carried out. To simplify, the decomposition is fixed, because of consequences on the incomes of the agents. We suppose moreover than there is no constraint on the sub-tasks order.
The objective of the proposed algorithm is to allow free agents having potentially incompatible interests to find a consensus on the distribution of the sub-tasks, but from their point of view.
Agents's motivations to carry out the sub-tasks are the associated profits, but the proposed protocol takes only preferences into consideration.

4 Concepts

4.1 The problem concepts

We now present the concepts of the problem and show its meanings within an example : airlines choose to cooperate to propose to their passengers a unified reservation system. The problem is that for each travel, several airlines are in competition on some stages.

Definition 1 **: Coalition Formation Problem (CFP)**
A CFP is defined by $\langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{G} \rangle$, where :
$\mathcal{A}$: the set of agents candidates to the execution of sub-tasks ;
$\mathcal{T}$: the set of tasks which will have to be accomplished ;
$\mathcal{S}$: the set of sub-tasks to be carried out ;
$\mathcal{C}$: the set of competences necessary to the sub-tasks ;
$\mathcal{G}$: the set of incomes.
Formally :
Definition 2 **: Agent**
An agent $a \in \mathcal{A}$ have a certain set of competences ; an agent $a \in \mathcal{A}$ is thus partially defined by : $a = \langle C, \ldots \rangle, C \subset \mathcal{C}$. This definition will be supplemented later by other elements, like the strategy (see (5.3, p. 5)).

Example $\mathcal{A} = \{EUropeanAirlines, AMerican\text{–}Airlines, WOrldAirlines, USAirlines, AFrica\text{–}Airlines, FRanceAirlines, BUsinessAirlines\}$

Definition 3 **: Task**
A task $t \in \mathcal{T}$ is only defined by the set of sub-task which it contains : $t = \langle S \rangle, S \subset \mathcal{S}$.

Example A task is a flight between two cities which puts into others : $\mathcal{T} = \{$New York-MAdrid (via PAris and LYon), Los Angeles-MOscow (via New York and PAris) and BErlin-JOhannesburg (via PAris)$\}$.

Definition 4 **: Sub-task**
A sub-task $s \in \mathcal{S}$ is defined by $s = \langle C, g \rangle, C \subset \mathcal{C}, g \in \mathcal{G}$, where c is the set of competences which an agent must have to be able to carry out the sub-task, and g the associated profit. This profit will be used by agents to compute his preferences.

Example $\mathcal{S} = \{$New_York$\rightarrow$Paris, Lyon$\rightarrow$Madrid, Paris$\rightarrow$Moscow, $\ldots\}$. We can now define the task $NY\!-\!M$ by $NY\!-\!M = \langle \{$NY$\rightarrow$P, P$\rightarrow$L, L$\rightarrow$M$\}, \ldots \rangle$.

Definition 5 **: Competence**
A competence $c \in \mathcal{C}$ is a single item which represents what is required to be carry out by an agent. A sub-task can require more than one competence.

Example Each flight needs competences : authorization to do a national stage (autFrance, autUSA,...), passengers capacity (Weak, Middle, Large), range of action(Very-Short-Range, Short-Range, Medium-Range, Long-Range).
$\mathcal{C} = \{autX, WC, MC, LC, VSR, SR, MR, LR\}$
$EUA = \langle \{autFR, autEU, autRU, WC, MC, SR, MR\} \rangle$

Definition 6 **: Profit**
A profit $g \in \mathcal{G}$ is used as an income, but only to simplify agents's internal calculations : $\mathcal{G} = [0, ProfitMax]$. However, the type of profits independence implies that any unit could have been used.

Example We choose $\mathcal{G} = [0, 10000]$ and for example, we have $NY - M = \langle \{NY\rightarrow$P, P$\rightarrow$L, L$\rightarrow$M$\}, 8000 \rangle$.

4.2 The resolution concepts

To solve the problem, agents exchange their preferences about possible solutions. If no consensus is reached, they can form an alliance. So, agents need to represent solutions, preferences, alliances and coalitions.
To reach a consensus, agents have to modify their opinion. Among all the imagined possibilities (pressure, corruption...), we choose the preferences exchange. First, it appears more rational to us, thus leads to a more legitimate solution. Then it can be used by heterogeneous agents, when messages using high level languages need complex symbolic process.
Definition 7 **: Solution**
A solution is an assignment of each sub-task to an agent which is able to perform it. A solution $\sigma \in \Sigma$ is an application $\mathcal{S} \rightarrow \mathcal{A}$ such that $\forall s \in \mathcal{S},\ a = \sigma(s) \Rightarrow s.C \subset a.C$.

Example $\sigma_{15} = [$NY$\rightarrow$PA$_2\hookrightarrow$WOA, L$\rightarrow$M$\hookrightarrow$BUA, P$\rightarrow$MO$\hookrightarrow$EUA, BE$\rightarrow$P$\hookrightarrow$FRA, LA$\rightarrow$NY$\hookrightarrow$USA, P$\rightarrow$JO$\hookrightarrow$AFA, NY$\rightarrow$PA$_1\hookrightarrow$WOA, P$\rightarrow$L$\hookrightarrow$FRA$]$.

Definition 8 **: Preference**
A preference is represented by distances (not in mathematical meaning) $\delta \in \Delta$ between solutions, where $\delta : \Sigma \times \Sigma \rightarrow [-1, 1]$ is an antisymmetrical application. So, $\delta(\sigma_1, \sigma_2) = d$ is interpreted by "σ_2 is preferred to σ_1 with a distance d if $d \geq 0$ and σ_1 is preferred to σ_2 with a distance $-d$ if $d < 0$". A null distance means that the solutions are indifferent.

Example Let $S_1 = \{\sigma_0, \sigma_2, \sigma_4, \sigma_6, \sigma_8, \sigma_{10}, \sigma_{12}, \sigma_{14}\}$ the set of solutions which provide outcomes and $S_2 = \{\sigma_1, \sigma_3, \sigma_5, \sigma_7, \sigma_9, \sigma_{11}, \sigma_{13}, \sigma_{15}\}$ the set of solutions which provide none. $\delta(\sigma, \sigma') = 0$ if σ and σ' are in the same set, and $\delta(\sigma, \sigma') = 1$ otherwise.

Definition 9 **: History**
A history $h \in H$ is a sequence of sights v_t, where each sight $v_t \in V$ is an application $\mathcal{A} \rightarrow \Delta$ and t the turn number. A history $h = (v_t)_{1 \leq t \leq T}$ represents all preferences exchanged from the first turn to the last one.
An alliance is a set of agents and behave like a single one. A member have a representative role : he communicates with outside agents (sending alliance

preferences and criteria).
Definition 10 **: Alliance**
An alliance $\lambda \in \Lambda$ is defined by $\lambda = \langle A, a_{rep} \rangle$, where $A \subset \mathcal{A}$ and $a_{rep} \in \mathcal{A}$ an alliance member with a special role, with the constraint that an agent can belong to only one alliance. The agent a_{rep} can be unknown ; in this case, it will be noted "?".

Example $\lambda = \langle \{USA, EUA, BUA\}, EUA \rangle$.

Definition 11 **: Coalition**
A coalition $\Xi(\sigma, t) \subset \mathcal{A}$ associated to the task $t \in \mathcal{T}$ in the solution $\sigma \in \Sigma$ is defined by : $\Xi(\sigma, t) = \{a \in \mathcal{A} / \exists s \in \mathcal{S}, s \in t.S, \sigma(s) \ni a\} = \bigcup_{s \in t.S} \sigma(s)$. A coalition contains all the agents which take part in a task.

5 Procedures

5.1 Common procedures

These functions have to be used by all agents.

- Alliance Fusion
- Alliance Preferences Computation

Definition 12 **: Alliance Fusion**
An alliance fusion operation $fusion : 2^{\Lambda} \rightarrow \Lambda$ is defined by $fusion(\{\lambda_1, \ldots, \lambda_n\}) = \langle A, a_r ep \rangle$, $A = \bigcup_{i \in [1,n]} \lambda_i.A$ and $a_{rep} = election(A)$, where λ_i are alliances who wish to form a new alliance and *election* a procedure of election of an agent among alliances members.
These operation can be applied to agents by converting an agent a to an alliance $\langle \{a\}, a \rangle$.

5.2 Alliance members procedures

Since an alliance has the same behavior than an agent, it must do similar computations. We have two possibilities : the first one is to define a function which use agents preferences or criteria to compute one preference or criterion ; the second is to define a function at alliance level which use agents basic criteria (*i.e.* the criteria they used to compute preferences and criteria). In these two cases, the agent with representative role take informations from all alliance members and compute the alliance preference or criterion.
In this second solution, all agents must have the same basic criteria, what restricts possibilities and limits evolution. In both cases, a function is imposed, though parameters can be negotiated, but the choice of this function has more influence in the first solution than in the second. In fact, it's difficult to find in the first case a function which behave according to our intuition (that was suggested by Kenneth Arrow in [1]). We still choose this last solution.
Definition 13 **: Alliance Preferences Computation**
An alliance preferences computation APC is an application $\Lambda \rightarrow \Delta$. This application is known only by alliance members ; other agents only known result of computation. To simplify, we use the same application for all agents, but the algorithm doesn't deal with it : alliance looks like a black box for the agents out of it.

Example Let $\lambda \in \Lambda$ an alliance, $\lambda = \{A, a_r ep\}$, $A \subset \mathcal{A}$. $APC(\lambda) = \delta$, where δ is defined by : $\forall (\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = \sum_{a \in A} a.\delta(\sigma_1, \sigma_2)$. This example use only members preferences to compute alliance preference.

5.3 Agent procedures

Each agent has his own strategy which can be applied using internal procedures :

- Independent Preferences Computation : computation of the first preferences without knowing those of the others.
- Dependent Preferences Computation : computation of preferences of next turns.
- Releasing Switch-over Proposal Criterion : criterion used to decide when to propose to release to switch-over mode.
- Releasing Switch-over Acceptance Criterion : criterion which decides to accept or not to switch to release mode.
- Alliance Formation Proposal Criterion : gives a list of agents to which to propose to form an alliance.
- Alliance Formation Acceptance Criterion : allows to answer to alliance formation propositions.

Since these functions are strategy dependent, we can only give formal definitions and examples, but no general formulation.
Definition 14 **: Independent Preferences Computation**
An IPC is an element δ of Δ.

Example Let $\delta = IPC$, $\forall(\sigma_1, \sigma_2) \in \Sigma^2$), $\delta(\sigma_1, \sigma_2) = profit(\sigma_2) - profit(\sigma_1)$. δ is an antisymmetrical application.

Definition 15 **: Dependent Preferences Computation**
A DPC is a function $H \to \Delta$, $h \mapsto \delta$.

Example Let $\delta = DPC(h)$, $h = (v_t)_t$. $\forall(\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = [\sum_{a \in \mathcal{A}}(v_T(a))(\sigma_1, \sigma_2)]/|\mathcal{A}|$. δ is an antisymmetrical application.

Definition 16 **: Releasing switch-over proposal criterion**
A $RSPC$ is an application $H \to \{False, True\}$.

Example Let $h = (v_t)_{1 \le t \le T}$. $RSPC(h) = False$ if $T \le 2$ and $RSPC(h) = (v_T = v_{T-1}) \vee (v_{T-1} = v_{T-2}) \vee (v_T = v_{T-2})$ otherwise. To diminish computation complexity, only loops of length 3 or less are detected.

Definition 17 **: Releasing switch-over acceptance criterion**
A $RSAC$ is an application $H \to \{False, True\}$.

Example $RSAC = RSPC$

Definition 18 **: Alliance formation proposal criterion**
An $AFPC$ is an application $h \mapsto (\lambda_1, \dots, \lambda_n)$, where $AFPC(h) = \emptyset$ is allowed and is interpreted by "agent doesn't want to form an alliance".

Example Let $d : \Delta \times \Delta \to \mathbb{R}$ a distance between agents's preferences, for example : $\forall(\delta_1, \delta_2) \in \Delta^2$, $d(\delta_1, \delta_2) = \sum_{(\sigma_1,\sigma_2) \in \Sigma^2} |\delta_1(\sigma_1, \sigma_2) - \delta_2(\sigma_1, \sigma_2)|$. For an agent a, $AFPC(h)$ is the set of agents which preferences are enough near to him using a threshold.

Definition 19 **: Alliance formation acceptance criterion**
An $AFAC$ is an application $H, A \to \{False, True\}$.

Example We can use the same application than the one above but using a greater threshold.

5.4 Parallel diffusion

In [6], agents broadcast their encoded informations, then when they have received encoded informations from all the others, they diffuse the key. Using this diffusion, agent can cheat : in a system with three agents a b and c, a sends its preferences to b and c, b to a and c, but c only to a ; a has received all the preferences, it sends its key to b and c, which makes it possible c to send to b another preference.
The authors supposed that agents did not cheat too much ; we improve this mechanism to prevent any fraud sending and receiving information in the same process. Agents send their encoded preferences with a private key, then when each one has receipted all the preferences, he diffuses an acknowledgment of delivery. Each agent diffuses his key only when he has received all acknowledgments. This diffusion solves the problem, because if one carries out the same scenario here, the cheating agent will not be able to change its encoded message any more. It could of course send a wrong key, but the fraud would be visible.

Algorithm 1 Parallel Diffusion of a data θ in $\mathcal{A}$

```
for all $a \in \mathcal{A}$ do
    $\theta^* \leftarrow Encrypt(\theta, key)$
    $\mathcal{B} \leftarrow \mathcal{A} \setminus \{a\}$
    $a$.broadcast($\theta^*$,$\mathcal{B}$)
    $a$.receipt($\theta^*$,$\mathcal{B}$)
    $a$.broadcast($Ack$,$\mathcal{B}$)
    $a$.receipt($Ack$,$\mathcal{B}$)
    $a$.broadcast($key$,$\mathcal{B}$)
end for
```

5.5 Agents's roles

Each agent can play several roles within the system :

- Organizer role : he is in charge of the management of inscriptions, turns and sendings of datas ;
- Candidate role : he seeks sub-tasks to be carried out. He receives and sends his preferences within his alliance or the system, proposes/accepts the switch-over releasing mode, proposes/accepts the alliance formation ;
- Supervisor role : he is responsible for checking respect of the protocol, in particular that agents send same information to all others ;
- Representative role : he receives the preferences from other members of the system and broadcasts it to internal ones and reciprocally.

6 Algorithm

We will only describe the candidate's algorithm, because it plays a leading role. Each agent has a list of interlocutors $InterList \subset \mathcal{A}$ initialized with the list of the candidates. The following algorithm is carried out by each agent a_i in a distributed way.

Algorithm 2 Switch-over mode

```
broadcast("Propose to form an alliance", $AFPC(h)$)
for $a$ such that receive("Propose to form an alliance",$a$) do
  if $a \in AFAC(h)$ then
    {according to $AFAC$, $a$'s proposition is accepted}
    send("Accept to form an alliance",$a$)
  end if
end for
if no formed alliance then
  system selects entities with nearest preferences
  system force them to form an alliance
end if
Bring up to date $InterList$
Return.
```

Algorithm 3 Main

```
$IndPref \leftarrow IPC$ {Computation of the independent preferences}
$h \hookleftarrow ParallelDiff(IndPref, InterList)$
while the consensus isn't reached do
  if $RSPC(h)$ then
    {according to this criterion}
    send("proposition to switch-over mode",*)
  end if
  if receive("proposition to switch-over mode",?) then
    if $RSAC$ then
      {according to this criterion}
      send("proposition to switch-over mode",*)
    end if
  end if
  if $\forall a \in InterList$, receive("proposition to switch-over mode",a) then
    {switch-over mode accepted}
    call switch-over mode
  end if
  $DepPref \leftarrow DPC(h)$ {Computation of the dependent preferences}
  $h \hookleftarrow ParallelDiff(DepPref, InterList)$
end while
```

7 Algorithm analysis

7.1 Termination

Without assumption on the criteria of switch-over mode releasing, we are not able to guarantee that the process will terminate. However, if we made the assumption that the criterion consists in checking that there is no loop, we can prove that it ends.

Definition 20 **: A loop in a history**
We say that a history $h = (v_t)_{1 \le t \le T}$ contains a loop if $\exists(\tau_1, \tau_2) \in [[1, T]]^2$, $\tau_1 \neq \tau_2$ such that $v_{\tau_1} = v_{\tau_2}$.

Definition 21 **: A** CFP **detects loops**
A CFP $\langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{G} \rangle$ detects loops if (h contains a loop $\Rightarrow$ ($\exists a_0 \in A$ such that $a_0.RSPC(h) = True \wedge \forall a \in A$, $a.RSAC(h) = True$)). In other words, a CFP detects lopps if at least one agent detects it and all then accept to change mode.

Theorem 7.1 *If the CFP detects loops, then the program terminates.*

Proof As number n of agents and number k of solutions aren't infinite, there is only $n.k!$ possible sights. After $n.k! + 1$ turns, the history will necessarily contain two identical sights : thus, there will have been a loop. However a switch-over mode releasing leads directly or indirectly (*via* the forced releasing) to a formation of alliance. Whatever the consensus criterion, it is reasonable to suppose that if all agents agree, consensus is reached. However the formation of alliances leads at worst to a great alliance (which contains all agents), which will require $n - 1$ stages. Finally, after $(n.k! + 1)(n - 1)$ turns at worst, there is consensus. ∎

7.2 Complexity

Complexity depends in particular on the number of possible solutions which is directly related to the problem datas. Let us suppose that our system contains n agents. There's no sense to calculate the average number of agents able to solve a subtask, because that ignores either very widespread competences, and especially competences which are linked. Thus let us suppose that each agent can process a portion of $1/m$ of the tasks, thus a task has on average n/m agents that are able to solve it, what then gives $k = (n/m)^s$ solutions. When the number of agents grows, the quantity of solutions becomes too big to be processed ; it is then necessary to limit space to be considered, but there is then loss of information. In the most general case, our algorithm does not make it possible to change class of complexity, but a calculation on average shows that the more agents have competences which overlap, the more there is competition and the more the consensus is difficult to reach. However, it is reasonable to suppose that a well conceived system distributes competences in a uniform way. For a good complexity, the ratio n/m must be as smallest as possible.

7.3 Implementation

The objective is to check the implementability of the algorithm and to test the various parameters of the coalition formation : individual strategies of the agents (calculation of the preferences, criterion of releasing, formation of alliances) and of the global parameters (consensus criterion, choice of forced alliances).
We choose the Java language because it allowed *via* threads to naturally simulate the distributed characteristic of the M.A.S. Thus, each sending of message generates a thread, which makes it possible to simulate asynchronous communications.
Each agent has access only to its local data and can thus exchange information only *via* messages, which is in agreement with its autonomy. Each role is established in the form of a thread charged to react to the received messages or to act of its own head. When an agent receives a message, he directs it towards the thread charged to manage messages intended for the role.

8 Conclusion

From the point of view of deployment of M.A.S. in an economic context, it is necessary to consider heterogeneous, egoistic and free agents (reasoning and behavior). To arrive, within this framework, to form coalitions, we proposed a protocol based on an exchange of preferences computed with quantitative and qualitative criteria since dependent on the specific strategies to the agents. Moreover, we set up releasing procedures thanks to the flexible concept of alliance to avoid the system paralysis.
We showed that such a protocol is open, distributed and egalitarian. So it opens many prospects. First, we will seek to improve complexity of calculation by introducing heuristics. We will then be able to test the various parameters and to thus choose the most interesting criteria and strategies.
As in [3], it would be also interesting to make it possible the agents to keep a track of alliances which were beneficial, in order to accelerate convergence towards a consensus. Building a model of the others will make possible to integrate the concept of confidence and will limit suspicion.

References

[1] Kenneth ARROW. *The Origins of the Impossibility Theorem*, chapter 1, pages 1–4. Elsevier Science Publishers B. V., Amsterdam, 1991.

[2] Steven KETCHPEL. Forming coalition in the face of uncertain rewards. In *Proceedings of the 12th National Conference on Artificial Intelligence. Volume 1*, pages 414–419, Menlo Park, CA, USA, July 31–August 4 1994. American Association for Artificial Intelligence, AAAI Press.

[3] Jorge Anacleto LOUÇÃ, Helder COELHO, and Suzanne PINSON. Forming coalition in task oriented multi-agent systems. In *Proceedings Of IBERAMIA 1996*. IBERAMIA, October 1996.

[4] Ohn SHEHORY and Sarit KRAUS. Task allocation via coalition formation among autonomous agent. In Chris S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 655–661, San Mateo, August 20–25 1995. The International Joint Conferences on Artificial Intelligence, Morgan Kaufmann.

[5] Onn SHEHORY and Sarit KRAUS. Formation of overlapping coalitions for precedence-ordered task-execution among autonmous agents. In Victor Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems*, pages 330–337. International Conference on Multi-Agent Systems, MIT Press, 1995.

[6] Gilad ZLOTKIN and Jeffrey S. ROSENSCHEIN. Coalition, cryptography, and stability : Mechanisms for coalition formation in task oriented domains. In *Proceedings of the 12th National Conference on Artificial Intelligence. Volume 1*, Distributed AI, pages 432–437, Menlo Park, CA, USA, July 31–August 4 1994. American Association for Artificial Intelligence, AAAI Press.